Machine Learning

CSE 343/543

Lecture 3

Classification – Naïve Bayes, k-NN

Slide Credits: John Rose, USC; David Sontag, NYU;

Today’s Lecture

- Classification
  - Probabilistic
  - Nonparametric
  - Nonmetric

Probabilistic Classifier: Bayesian

- For Binary Classification, we are interested in

$$p(y|\boldsymbol{x}) \qquad\qquad y \in \{0,1\}$$

- Applying Bayes Rule

$$p(y|\boldsymbol{x}) = \frac{p(\boldsymbol{x}|y)p(y)}{p(\boldsymbol{x})}$$

- A Bayesian Classifier $\widehat{y}_i = f(\boldsymbol{x}_i, \alpha)$ will try to match the training data distribution

$$p(f(\boldsymbol{x}_i, \alpha)|\boldsymbol{x}_i; \alpha) \sim p(y_i|\boldsymbol{x}_i; \alpha)$$

Probabilistic Classifier: Bayesian

- Classification is then performed as
  - If $p(f(\boldsymbol{x}_i, \alpha) = 1|\boldsymbol{x}_i, \alpha)) > p(f(\boldsymbol{x}_i, \alpha) = 0|\boldsymbol{x}_i, \alpha))$

    then the label

    $$\widehat{y}_i = 1$$

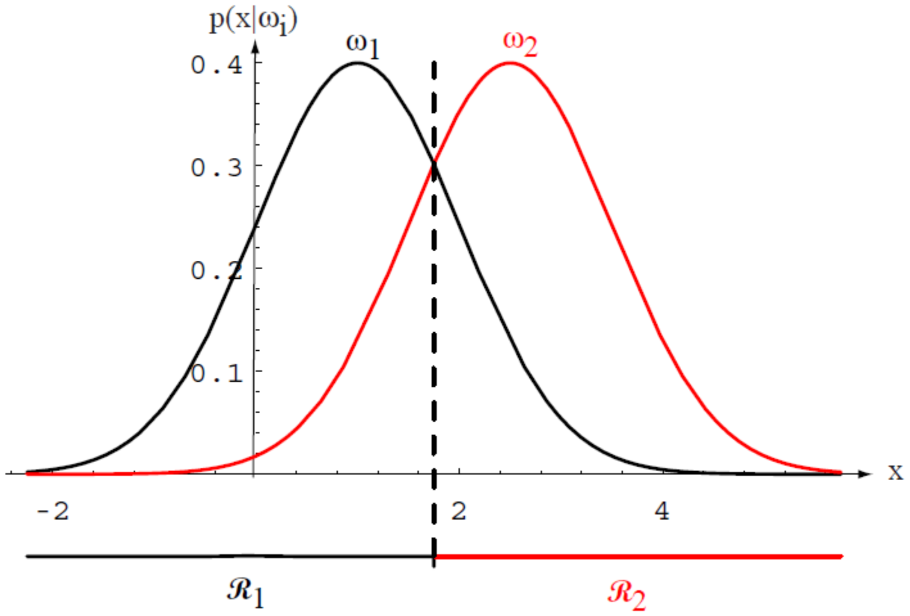
    else

    $$\widehat{y}_i = 0$$

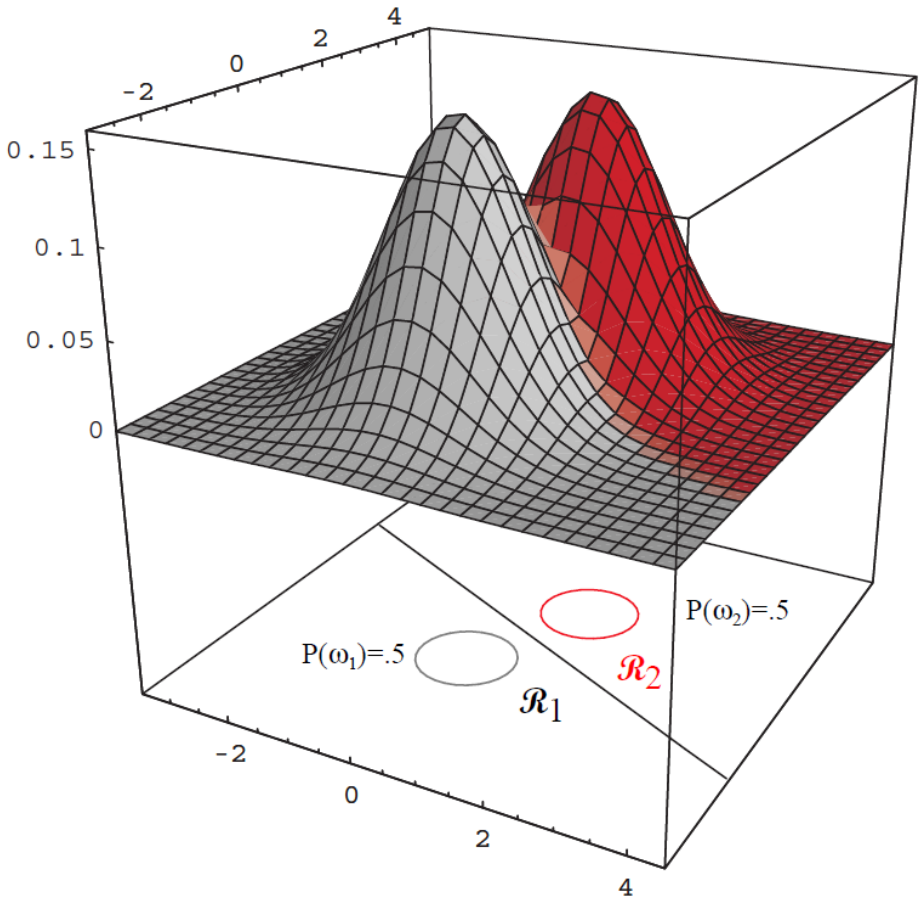
- For M-class classification

$$\widehat{y}_i = \arg \max_{y=\{0,1,\ldots,M\}} p(f(\boldsymbol{x}_i, \alpha) = y|\boldsymbol{x}_i)$$

Visualization: Bayes Classifier



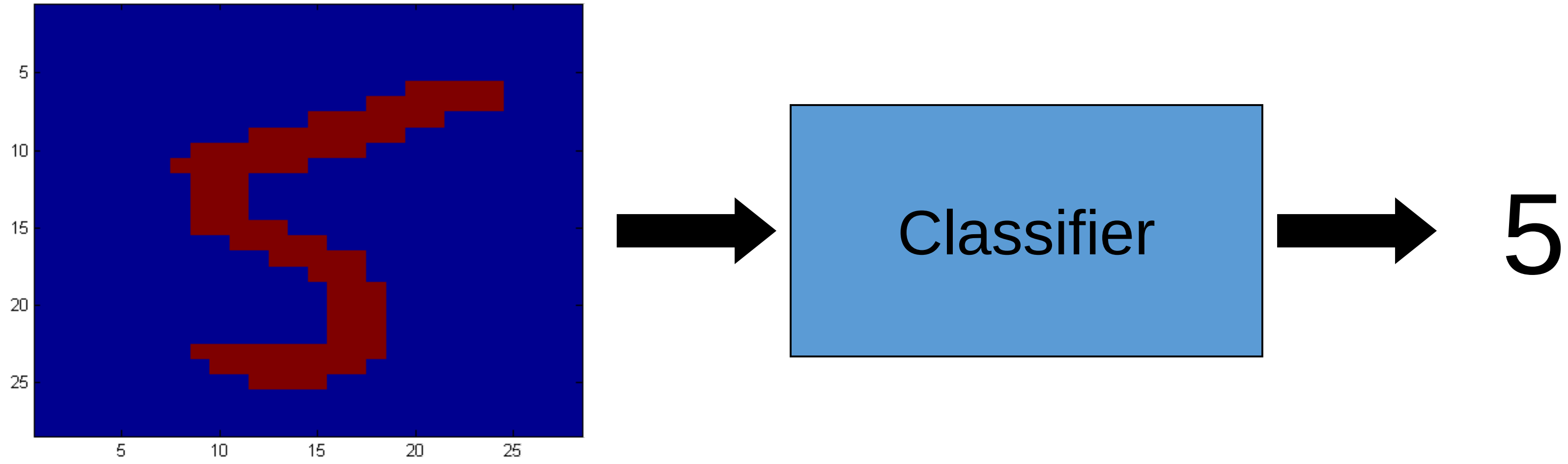
1-D



2-D

Concrete Example: Bayes Classifier

- **Digit Recognition**



- **$X_1,\ldots,X_n \in \{0,1\}$ (Black vs. White pixels)**
- **$Y \in \{5,6\}$ (predict whether a digit is a 5 or a 6)**

The Bayes Classifier

- We saw that a good strategy is to predict:

$$\arg\max_{Y} P(Y|X_1, \ldots, X_n)$$

  - (for example: what is the probability that the image represents a 5 given its pixels?)

- So … How do we compute that?

The Bayes Classifier

- Uses Bayes Rule!

Likelihood

Prior

Posterior

$$\underbrace{P(Y|X_1,\ldots,X_n)}_{\text{Posterior}} = \frac{\overbrace{P(X_1,\ldots,X_n|Y)}^{\text{Likelihood}}\overbrace{P(Y)}^{\text{Prior}}}{\underbrace{P(X_1,\ldots,X_n)}_{\text{Normalization Constant}}}$$

Normalization Constant

- Why did this help?
  - Well… We think that we might be able to specify how features are “generated” by the class label

The Bayes Classifier

- Let's expand this for our digit recognition task:

$$
\begin{aligned}
P(Y=5|X_1,\ldots,X_n) &= \frac{P(X_1,\ldots,X_n|Y=5)P(Y=5)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)} \\
P(Y=6|X_1,\ldots,X_n) &= \frac{P(X_1,\ldots,X_n|Y=6)P(Y=6)}{P(X_1,\ldots,X_n|Y=5)P(Y=5)+P(X_1,\ldots,X_n|Y=6)P(Y=6)}
\end{aligned}
$$

- To classify, we'll simply compute these two probabilities and predict based on which one is greater

Model Parameters

- For the Bayes classifier, we need to “learn” two functions, the likelihood and the prior

- How many parameters are required to specify the prior for our digit recognition example?

Model Parameters

- How many parameters are required to specify the likelihood?
  - (Supposing that each image is 30x30 pixels)

- # of parameters for modeling $P(X_1,\dots,X_n|Y)$:
  - $2(2^n-1)$

- The problem with explicitly modeling $P(X_1,\dots,X_n|Y)$ is that there are usually way too many parameters:
  - We'll run out of space
  - We'll run out of time
  - And we'll need tons of training data (which is usually not available)

The Naïve Bayes Model

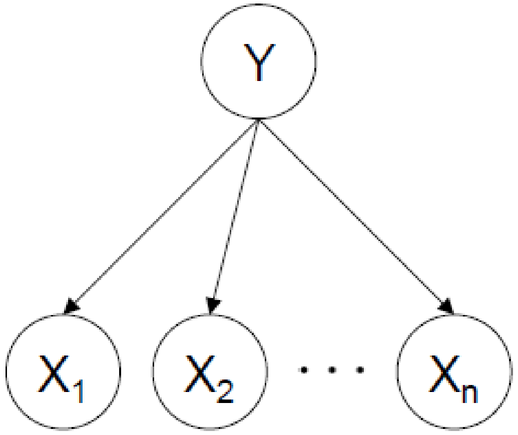
- The *Naïve Bayes Assumption*: Assume that all features are independent **given the class label Y**
- Equationally speaking:

$$P(X_1, \ldots, X_n | Y) = \prod_{i=1}^{n} P(X_i | Y)$$

- # of parameters for modeling $P(X_1|Y),\ldots,P(X_n|Y)$
  - 2n

The Naïve Bayes Classifier

- Given:
  - Prior P(Y)
  - *n* conditionally independent features **X** given the class Y
  - For each X_i, we have likelihood $P(X_i|Y)$



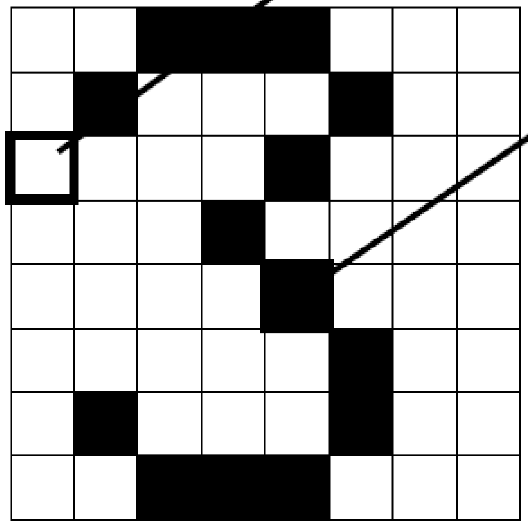
- Decision rule:

$$\begin{aligned}\hat{y} &= \arg\max_y P(y)P(x_1,\ldots,x_n \mid y)\\ &= \arg\max_y P(y)\prod_i P(x_i|y)\end{aligned}$$

What has to be learned?

$P(Y)$

1	0.1
2	0.1
3	0.1
4	0.1
5	0.1
6	0.1
7	0.1
8	0.1
9	0.1
0	0.1



$P(F_{3,1} = on|Y)$

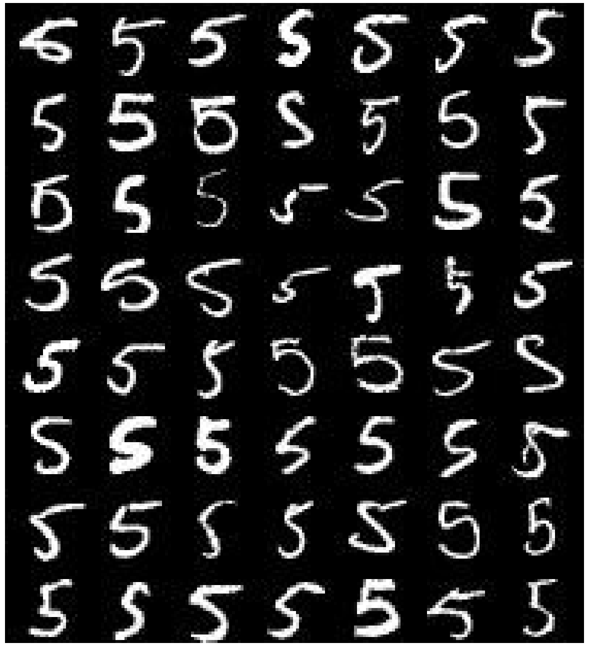
1	0.01
2	0.05
3	0.05
4	0.30
5	0.80
6	0.90
7	0.05
8	0.60
9	0.50
0	0.80

$P(F_{5,5} = on|Y)$

1	0.05
2	0.01
3	0.90
4	0.80
5	0.90
6	0.90
7	0.25
8	0.85
9	0.60
0	0.80

Naïve Bayes Training

- Now that we've decided to use a Naïve Bayes classifier, we need to train it with some data:



MNIST Training Data

Naïve Bayes Training

- Training in Naïve Bayes is **easy**:
  - Estimate P(Y=v) as the fraction of records with Y=v

$$P(Y = v) = \frac{Count(Y = v)}{\#\ records}$$

  - Estimate P(X_i=u|Y=v) as the fraction of records with Y=v for which X_i=u

$$P(X_i = u|Y = v) = \frac{Count(X_i = u \wedge Y = v)}{Count(Y = v)}$$

- (This corresponds to Maximum Likelihood estimation of model parameters)

Naïve Bayes Training

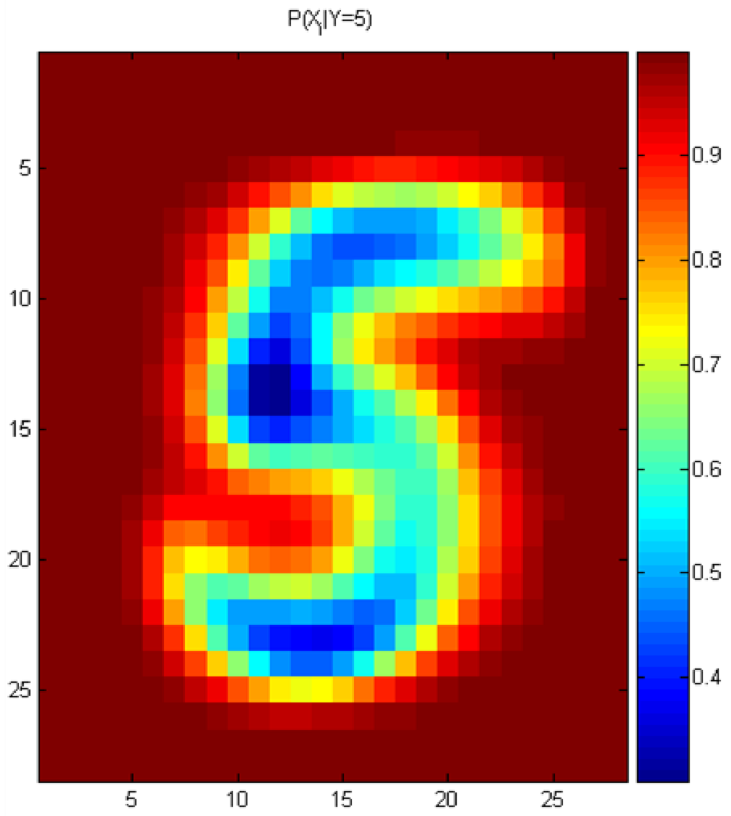
- In practice, some of these counts can be zero
- Fix this by adding "virtual" counts:

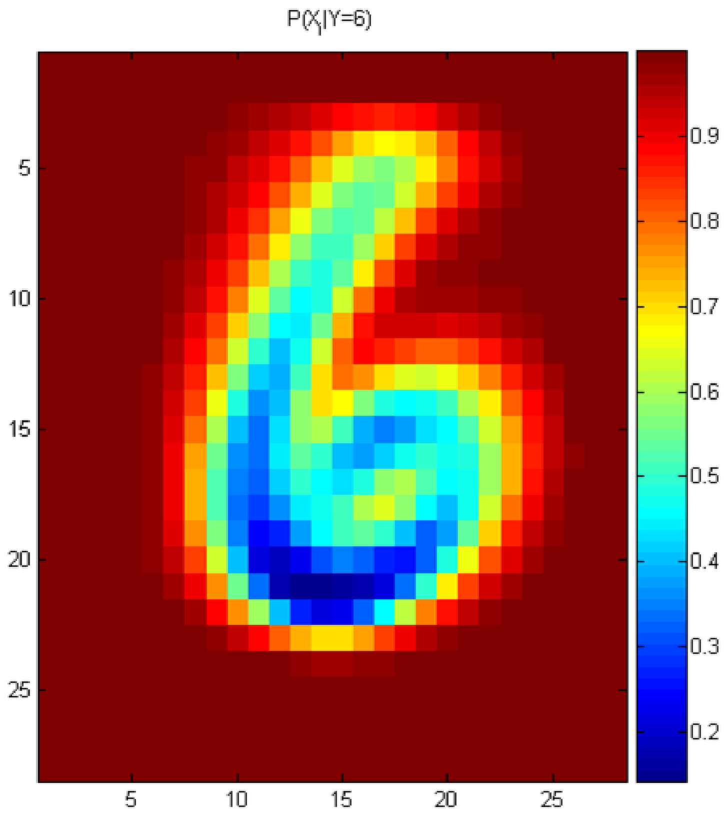
$$P(X_i = u|Y = v) = \frac{Count(X_i = u \wedge Y = v) + 1}{Count(Y = v) + 2}$$

  - (This is like putting a prior on parameters and doing MAP estimation instead of MLE)
  - This is called *Smoothing*

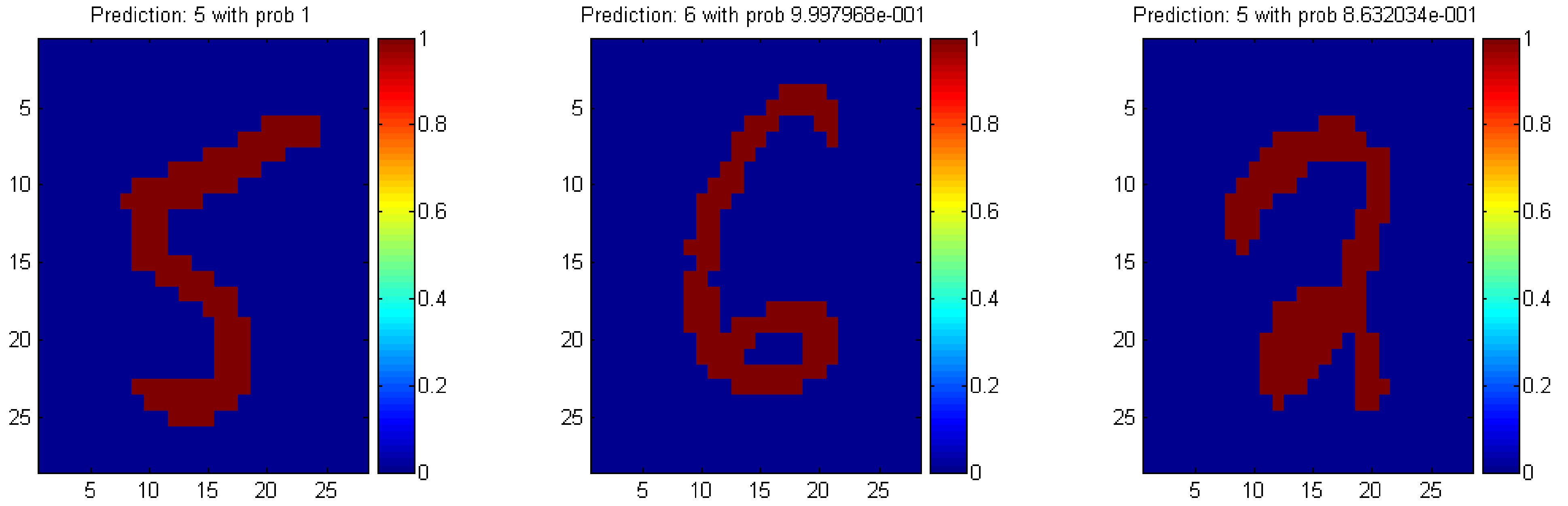
Naïve Bayes Training

- For binary digits, training amounts to averaging all of the training fives together and all of the training sixes together.





Naïve Bayes Classification

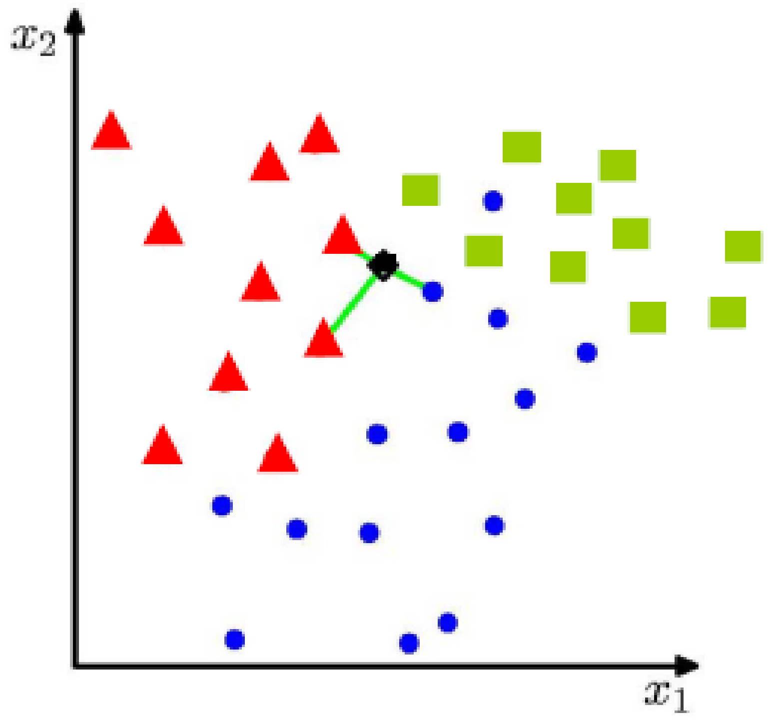


Non-parametric Classification: K-Nearest Neighbors

- Collect training data (x,y)

- For test sample
  - Find nearest neighbors in training set
  - Assign class label based on some <u>consensus</u>

- Parameters
  - Distance Metric for 'nearest' neighbors
  - Voting strategy



k-nearest neighbour (knn)

Non-metric Classification: Decision Trees